HARDWARE IMPLEMENTATION OF RE-CONFIGURABLE RESTRICTED BOLTZMANN MACHINES FOR IMAGE RECOGNITION

A Thesis
Presented to
The Academic Faculty

By

Soham Jayesh Desai

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in
Electrical and Computer Engineering

Georgia Institute of Technology

May 2015

HARDWARE IMPLEMENTATION OF RECONFIGURABLE RESTRICTED BOLTZMANN MACHINES FOR IMAGE RECOGNITION

Approved by:

Dr. Arijit Raychowdhury, Advisor
School of ECE
Georgia Institute of Technology

Dr. Sudhakar Yalamanchili
School of ECE
Georgia Institute of Technology

Dr. Sung Kyu Lim
School of ECE
Georgia Institute of Technology

Date Approved: April 21st 2015

ACKNOWLEDGEMENTS

While bringing out this thesis to its final form, I came across a number of people whose contributions in various ways helped my field of research and they deserve special thanks. It is a pleasure to convey my gratitude to all of them.

First and foremost, I would like to express my deepest sense of gratitude and indebtedness to my advisor, Professor Arijit Raychowdhury for his insightful guidance, encouragement and support throughout my thesis. Above all, his depth of knowledge in broad areas of interest to the industry and his ability to interconnect these for research and innovation has been truly inspirational and something which I desire to emulate.

I would like to thank Dr. Mohammed Shoaib (Microsoft Research) and Dr. Charles Augustine (Intel Corporation) for their collaboration with our project and describing the current interests from an industry perspective, providing me with groundwork for the project.

I am also grateful to Professor Sudhakar Yalamachili and Prof. Sun Kyu Lim for agreeing to be part of my thesis Reading Committee, providing constructive criticism and helpful suggestions after reviewing my research summary, based on their vast experience.

I am thankful to all the past & present Members of Integrated Circuits & Systems Research Lab including my fellow lab-mates Abhinav Parihar, Ashwin Chintaluri, Anvesha Amravati and Samantak Gangopadhyay for their support, friendship and sharing their experiences and interesting research work over a coffee break.

Finally, I would like to thank my room-mates over the two years for the camaraderie and my parents for their continuous motivation and support despite not being physically present along with me during the last two years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

IoTs	Internet of Things
BWCs	Body-Worn Cameras
RBM	Restricted Boltzmann Machine
ANN	Artificial Neural Networks
NPCs	Neural Processing Cores
FPGA	Field Programmable Gate Array
FIFO	First-In First-Out Module
MCMC	Markov Chain Monte Carlo

SUMMARY

The Internet of Things (IoTs) has triggered rapid advances in sensors, surveillance devices, wearables and body area networks with advanced Human-Computer Interfaces (HCI). Neural Networks optimized algorithmically for high accuracy and high representation power are very deep and require tremendous storage and processing capabilities leading to higher area and power costs. For developing smart front-ends for 'always on' sensor nodes we need to optimize for power and area. This requires considering trade-offs with respect to various entities such as resource utilization, processing time, area, power, accuracy etc. Our experimental results show that there is presence of a network configuration with minimum energy given the input constraints of an application in consideration. This presents the need for a hardware-software co-design approach. We present a highly parameterized hardware design on an FPGA allowing re-configurability and the ability to evaluate different design choices in a short amount of time. We also describe the capability of extending our design to offer run time configurability. This allows the design to be altered for different applications based on need and also allows the design to be used as a cascaded classifier beneficial for continuous sensing for low power applications. This thesis aims to evaluate the use of Restricted Boltzmann Machines for building such reconfigurable low power front ends. We develop the hardware architecture for such a system and provide experimental results obtained for the case study of Posture detection for body worn cameras used for law enforcement. Our proposed system, implemented on a Xilinx Virtex 7 XC7VX485T

platform consumes a minimum power of 19.18 mW with an accuracy of 80% for the selected case study of Posture Detection.

CHAPTER 1

INTRODUCTION

The “Internet of Things” represents a paradigm shift in the interconnected world, leading to communication among various physical entities around us. At the same time these devices are expected to possess sufficient intelligence to be able to assimilate, analyze and process data. Constraints due to battery life, storage capacity, transmission costs make it imperative to have a smart front-end capable of making decisions regarding the relevance and importance of the image, before storing or transmitting it. Artificial Neural Networks inspired from biology have been shown to provide us with such a capability and has been an active area of research of the last decade. The neural network processing comprises of a huge number of parallel computations and this has led to implementations based on Graphics Processing Units (GPUs) or Field Programmable Gate Arrays (FPGAs) which are well suited for such computations. For being part of the “always on” systems, the neural network implementation at the very front end needs to be simple, and has to use the least resources possible while meeting the provided constraints so as to consume the least total energy. The same module needs to be capable of being reused for making further accurate classifications once the relevant image has been filtered out. This cascading of classifiers is imperative to meet the requirements of low power and avoiding unnecessary storage or transmission.

This thesis aims to design an engine for neural network based classification capable of being reconfigured at compile time by extensive parameterization and also provides the capability of being extended to provide run-time configurability. It also explores a case study of significant recent interest to the society and performs power estimations and design space explorations.

CHAPTER 2

BACKGROUND AND MOTIVATION

Artificial Neural Networks attempt to model the information processing capabilities of nervous systems [1]. Each node in the artificial neural network can be considered as a computing unit capable of transforming the provided inputs to an output based on the weights, bias and the non-linearity present similar to a synaptic neuron. Simplest form of layered architectures are those in which a set of computing units N is subdivided into L subsets N1, N2 …, N_L in such a way that only connections from units in N1 go to units in N2, from units in N2 to units in N3, etc. The input is provided to the N1 layer and N_L is connected to the output sites. Connections within the same layer and from higher to lower layer are forbidden. All other layers with no direct connection from or to the outside are called Hidden Layers as shown in the Fig. 1. The input is processed and relayed from one layer to the other, until the final result has been computed [1]. The weights and biases can be trained using gradient decent and back-propagation as described in [2]. Such training algorithms require the presence of labeled data and are termed as supervised learning algorithms.

Artificial Neural Networks have evolved over the last few decades both with respect to the representational power and complexity requiring much greater computational power and higher resources for training as well as for classification. Lot of importance has been given to achieving human-like accuracy with respect to object classification, speech recognition, character recognition etc. Deep Neural Networks Consider Fig. 2, the objective is to classify the input as belonging to the category that a Man is present. This category where the Man is present is a high-level abstraction with respect to the space of input images [3]. This high level category has a direct relation with the sensory data comprised of raw pixels of the input image. This relation can be

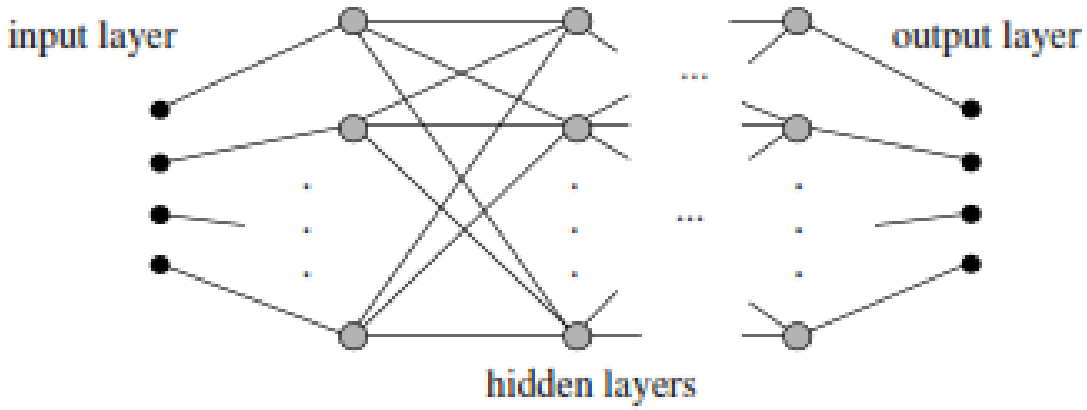


Fig. 1. Layered Architecture of Neural Networks. [1]

seen to be comprising of various intermediate and low level abstractions (for example: shape of the human face, or the height of the detected human object, shoulder width, etc.) which together can be used to detect the category of the man. Lower level abstractions are more directly tied to input pixel features, whereas higher level abstractions can be considered more abstract and closer to the category because their connection to actual input features is remote and via intermediate features. The focus of training the deep networks is to automatically discover such abstractions, from the lowest level features to the highest level concepts with little human intervention [3]. We thus want to train such a deep network without providing labeled data. Such a training is termed as an unsupervised training algorithm where the network tries to model the data without any labels. [4] Describes a learning algorithm which uses Restricted Boltzmann Machines to greedily train a layer at a time in an unsupervised manner followed by training in a supervised manner for the specific task. Training greedily in an unsupervised manner allows the network to find an internal representation of the network near a 'good' solution and has shown to give better results than training the network in a similar manner using a supervised manner with random initialization.

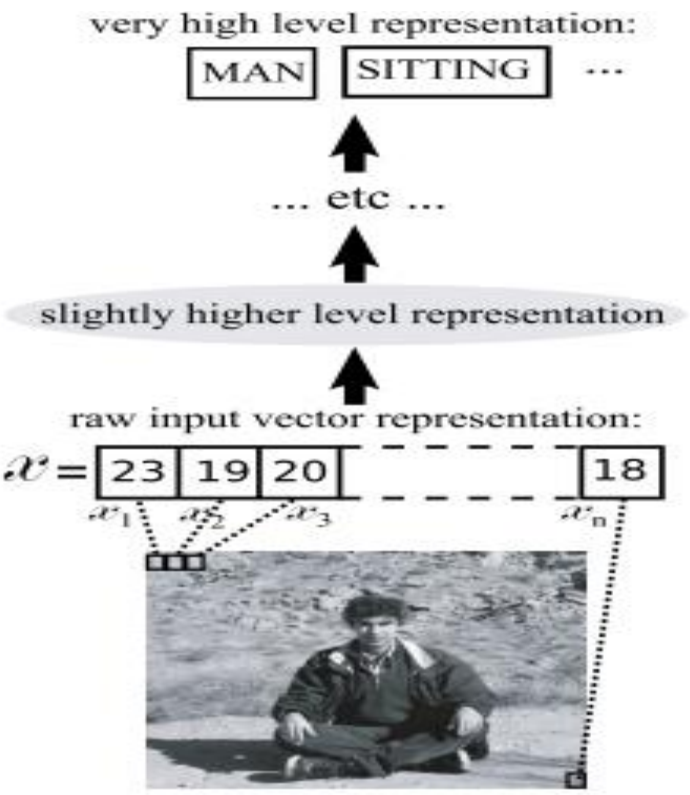


Fig. 2. The raw input image being transformed into gradually higher levels of representation consisting of more abstract functions of the raw input, e.g. Edges, local shapes, object parts, etc. [3]

Participants of competitions such as the Imagenet Large Scale Visual Recognition Challenge [5][6] show that currently deep learning and specifically deep neural networks provide highly accurate results with respect to the above applications even under acquisition noise and image occlusion.

However developing highly scalable and fully parallel hardware for such deep networks is not suitable for our application because: (1) such networks require tens to hundreds of thousands of neural processing units, or nodes which are typically executed in many-core servers and distributed machines (2) the power cost of such networks is prohibitive in a mobile platform and (3) they are not suitable for real-time applications. On the other hand, our accuracy targets for a network at the very front end can be relaxed

from that of deep networks (accuracies > 95-97%). We target > 80% accuracies (with minimum number of false rejects) but with tens of mW of power consumption. This is almost three orders of magnitude reduction of power when compared to deep networks optimized for performance and would enable true mobility.

Shallow networks such as Support Vector Machines with Gaussian kernel machines and single hidden layer Neural Networks have been shown to be able approximate any function with arbitrary precision. [7]. For determining just the relevancy of an image such a network can prove useful and efficient. We opt for Restricted Boltzmann Machines (RBM) (described ahead) rather than support vector machines as this allows us to reuse the same RBM engine for constructing a deeper network with more representational power. Hence, we adopt Restricted Boltzmann Machine based Artificial Neural Networks (ANNs) as the algorithmic and hardware design paradigm for ultra-low power recognition. Restricted Boltzmann Machine (RBM) based recognizers are probabilistic graphical models (which form the basis of deeper networks). RBMs are modular, scalable and can be efficiently mapped to hardware with well-controlled data movement between logic and embedded memory. RBMs allow us to re-use the same resources via time multiplexing because of their modularity and Single Instruction Multiple Data (SIMD) nature. We also provide an option of increasing the network depth, for potentially higher accuracy, which amounts to storing different sets of weights for each layer and reusing the available computational resources.

Mathematical Description

The basic RBM consists of two layers, an output visible layer “V” representing the observable data and a hidden layer “H” which portrays the internal representation of the observable data into the system. These layers are comprised of processing elements

referred to as Neurons or nodes. RBMs form a special category of Boltzmann Machines where these two layers form a bipartite graph. There are no connections between the hidden neurons. Each hidden unit describes a probability distribution over the inputs provided by the visible layer units. Further, the hidden layer provides a higher level of feature set for the input data and enables associativity between a set of observable outputs and control inputs. Using the following notation: $V = (V_1 \ldots V_m)$ representing the Visible input units, $H = (H_1 \ldots H_n)$ representing the Hidden Neurons, and the random variables V and H take binary values (v,h). The joint probability distribution for both the layers is given by the Gibbs Distribution [8].

$$p(v,h) \; \alpha \; e^{-E(v,h)} \tag{1}$$

Here the Energy function is given by

$$E(v,h) = -\sum_i \sum_j w_{ij} h_i v_j - \sum_j b_j v_j - \sum_i c_i h_i \tag{2}$$

The j and i sum over all the nodes in the visible layers and hidden layer respectively. w_{ij} represents real valued weights across the edge between the j^{th} visible node and i^{th} hidden node. b_i and c_j represent the real valued bias terms associated with the j^{th} visible node and i^{th} hidden node respectively.

Based on this energy it can be shown [8] that the conditional probability of any unit being 1 can be written as

$$P(V_j = 1 \mid h) = sig\left(\sum_i w_{ij} h_i + b_j\right) \tag{3}$$

Here "sig" refers to the sigmoid function. These equations show that an RBM can be reinterpreted as a standard feed-forward neural network with one layer of non-linear processing units.

Training of RBMs

The weights need to be modified such that the RBM produces the minimum energy across the training set of observable data. The accurate calculation of the log-likelihood gradient is computationally prohibitive. We follow the method provided in [8] for approximating the RBM log-likelihood gradient namely, "Contrastive Divergence" which was originally described in [9]. Obtaining unbiased estimates of the log-likelihood gradient using Markov Chain Monte Carlo (MCMC) methods typically requires many sampling steps. In [9] the authors show that estimates obtained after running the chain for just a few steps can be sufficient for model training. We follow the training algorithm described in [9] for training the RBM called Contrastive Divergence.

The main intuition or the main motivation behind this algorithm is that after a few iterations of performing MCMC, the data would have moved from the target distribution to the proposed distribution. The gives us an idea of direction in which the distribution should move to better model the data and also allows us to calculate an approximate gradient. Empirically, Hinton shows in [9] that even a single cycle of MCM is sufficient for the algorithm to converge.

CHAPTER 3

HARDWARE IMPLEMENTATION OF NEURAL NETWORKS

FPGA based reconfigurable computing architectures are well suited to implement ANNs as one can exploit concurrency and rapidly reconfigure to adapt the weights and topologies of an ANN suitable for quicker design space exploration [10]. In contrast to a Custom ASIC chip, the FPGAs are readily available at a reasonable cost and have a reduced hardware development cycle. The measurement results obtained for an FPGA can be interpolated to that of custom chip which would be the final objective after the completion of design space exploration.

Related Work

Significant Research has been conducted on creating a scalable FPGA or GPU based implementation of Restricted Boltzmann Machines with emphasis on training. Training of Neural Networks is known to be computationally expensive justifying the need of developing accelerators for such networks. This thesis specifically concentrates on designing and evaluating a low power accelerator for classification along with meeting the real time constraints. For example, [11] shows a speedup of 25-30 as compared to optimized C++ program running on high-end CPU specifically for training, [12] describes a technique to distribute a single RBM over multiple FPGAs so that a high number of neurons can be instantiated leading to high performance, also introducing the concept of virtualization or reuse for instantiating multiple RBMs using the same

resources, [13] which investigates the use of GPUs for training of RBMs and reports a speedup of 66 over a C++ program running on an 2.83 Ghz Intel Processor.

Using fixed point arithmetic provides improvements in processing time as well as energy for training and classification. [14] Evaluates the tradeoff of accuracy and processing time with respect varying fixed point representations and different piecewise linear approximations for the sigmoid unit for training.

[15] Focuses on robustness of Deep Belief Networks by introducing random errors during pre-training, training and testing. It also concludes that it's possible to have low precision implementations of the testing stage in fixed point with negligible loss in accuracy, which proves to be very useful for energy considerations and is adopted in this thesis.

[16] Follows a similar approach to that of this thesis. It describes a scalable processing core and has capability of on-line learning as a configurable parameter. The design presented in this thesis also makes use virtualization w.r.t layers and neurons and uses a distributed controller helping with simplification of re-use of resources. The network presented in the paper is relatively small from 10-3-1 to 10-6-3-2, the design presented in this thesis can be scalable to arbitrary sizes at the expense of processing time. This thesis also conducts an evaluation of different tradeoffs involved keeping emphasis on the total energy consumption.

Design of the Reconfigurable RBM Engine

To meet the extreme power constraints in 'always-on' sensor nodes, custom hardware architecture is required. We have implemented the proposed algorithm on a Xilinx Virtex 7 XC7VX485T platform shown in Fig. 3.

The motivation for RBM based ANNs comes from the models of synaptic behavior of human neurons, by computing the function:

$$\sum i \, Wip \, Xip + \theta p. \quad (4)$$

In Equation (6), W_{ip} represents the synaptic weights, X_i represents the input feature to the neuron, θ_i is the bias for the p^{th} neuron and these are summed over all the input features of the image. Each Neuron in our network models the computation represented by (4). We call the instantiation of this neuron in hardware as the neuron processing core (NPC) illustrated in Fig. 4. The NPC comprises of a fixed point signed multiplier, accumulator, and memory for storing the weights. The weights are stored as distributed memory within each neuron core. A hidden layer in a neural network comprise of many such neurons performing a similar computation as (3) but with a different set of weights and biases. Similarly, in hardware many such NPCs are grouped together to form a layer. The input to each layer is provided parallel to all the NPCs within the Layer. The inherent parallelism of such a neural network results from the fact that, in a fully connected network, the computation in (6) is carried out by all the neurons in parallel for an input feature Xi.

Fig. 3. Xilinx Virtex 7 VC707 Evaluation Kit. The Reference Board used for the Design And Experimentation

Resource Reuse or Virtualization

Ideally to obtain the least processing time we would desire as many NPCs in parallel as the number of neurons in the hidden layer. This results in very high resource utilization and consequently greater overall power and area. We provide the capability to reuse these NPCs by time multiplexing, for computations belonging to the same layer. We differentiate between these as “Virtual” and “Physical” NPCs. Physical NPCs are instantiated in the physical design and consume physical resources. This comes with large area and power (both leakage and dynamic). Virtual NPCs represent the actual number of neurons in a hidden layer for a particular network configuration. The ratio between Virtual and the Hidden Neurons gives you the number of “phases” or the number of times these NPCs need to re-execute so that the computation for the layer gets completed shown in Fig. 5. The concept of virtualization also proves useful for providing

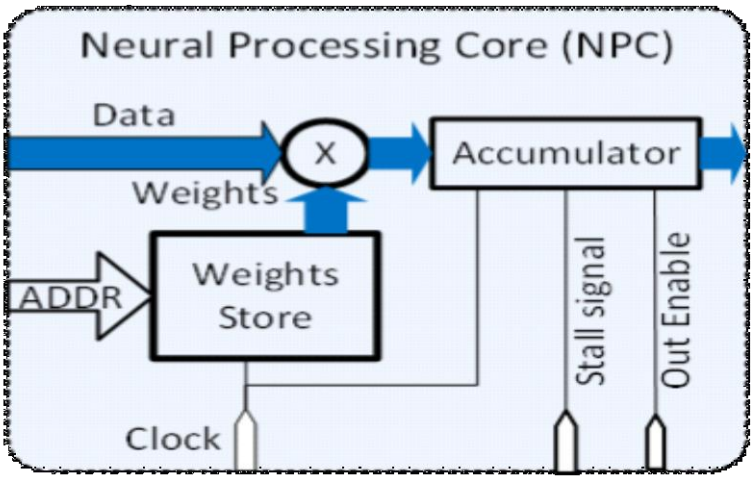


Fig. 4. Block Diagram of the Neuron Processing Core showing the incoming and outgoing control signals, the data path and the address bus

run-time configurability as described later in this report. The most serialized case comprises of a single NPC executing as many times as the number of virtual NPCs or neurons in the layer, resulting in the least amount of resource utilization, power but much higher processing time. In ‘always-on’ microphone-based audio sensors, serialization of parallel workload to a certain extent has been shown to be effective in reducing the overall system power. For a given computational complexity at a frame rate of 30fps, a lower number of ‘virtual NPCs’ demonstrate a favorable trade-off between power and the total computational time. The layer as described by Fig. 6

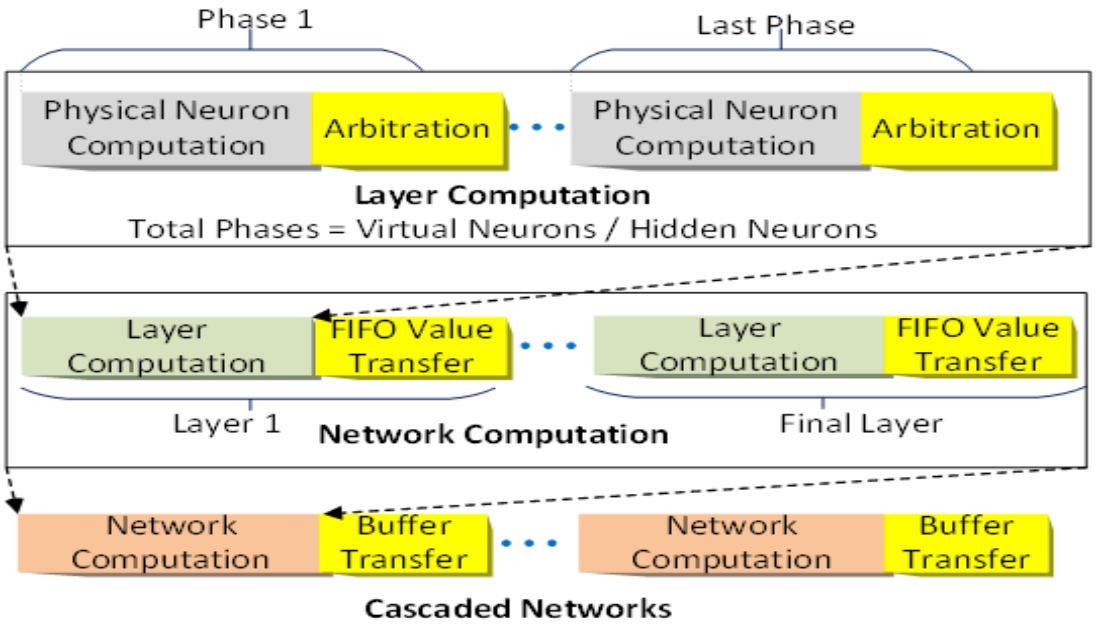


Fig. 5. Depicts the Virtualization of Neuron Computation, Layer Computation and the entire network. The Physical NPCs are reused for a count equal to Total Phases so as to compute for all the Virtual NPCs. The Layer can be reused to provide an increase in depth of the network. Similarly, The Entire Network can then be reused for a different purpose or even for recognizing the same image with higher Accuracy.

also consists of a sigmoid approximating unit, a control unit, a bus arbitration unit and a first-in, first-out (FIFO). Direct implementation of a sigmoid unit is expensive in hardware and increases the power and the processing time. We approximate the sigmoid using a piece-wise linear approximation. The sigmoid operation is common to all neurons and thus doesn't need to be part of the individual neurons. We incorporate the sigmoid unit before the input of the FIFO so as to ensure that all neuron outputs are being passed through the sigmoid non-linearity before proceeding to the next layer. We opt for a distributed control unit so that the computation of layers remain as independent from

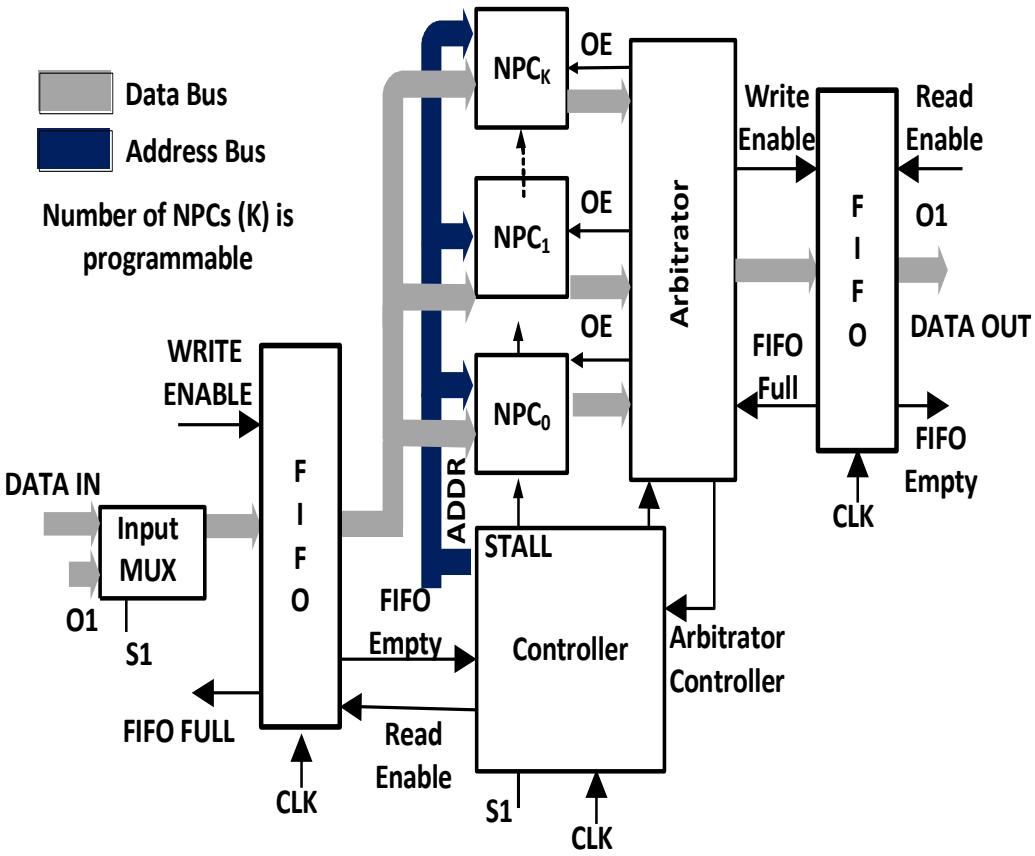


Fig. 6. The Block Design of a Single Layer showing the control and data flow. 01, the output of the layer if input back into a multiplexer. S1, a control signal from the Controlle is used for selecting the input.

each other as possible. The control unit provides the address for the weights, communicates with other layers and provides control signals to the NPCs, bus arbitration unit and the FIFO. A bus arbitration unit is required to serialize the neuron outputs generated and store it in the output FIFO, before the next computation of the layer can take place. We pipeline the layers using a FIFO. The FIFO full and empty signals are used for the communication between the layers. If any succeeding stage is still processing the data, the preceding layer is stalled from transferring the values from its output FIFO. The system is thus a pull-based pipelined system. To allow multiplexing the FIFO length equals the number of Virtual NPCs. Similar to the concept of re-using the NPCs, we

provide the capability of reusing the layer by allowing the output FIFO to feed data back as input. This path is multiplexed with the original input path. The end of the network consists of a final layer, also called as the class output layer which comprises of a store buffer, counter and a comparator in addition to the NPCs, FIFO and the control unit. The store buffer is used for storing the largest value read from the FIFO. The counter keeps track of the output number of the NPC because this corresponds to the classification label. Input is serially fed from the FIFO into the comparator and signed compared with the value stored in the store buffer. The store register and the counter are updated if the input value is greater than the store buffer. It is beneficial to keep final layer as parallel as possible, since it allows us to avoid the replay of outputs from the previous layer. The weights, input features and the data transferred are represented using a signed fixed-point notation Q2.10 for the base case. Fixed point data representation of resolution more than Q2.6 shows no impact on recognition accuracy over a floating point representation and results in significant cost savings with respect to resource utilization and power. The accumulator output buffer resolution is kept significantly greater than the resolution of the input to the accumulator to prevent any overflows which has shown to impact the accuracy of the network severely.

Compile Time Configurability

The entire design is made configurable by extensively parameterization. This allows us to perform design space exploration where the role of these parameters on performance, power and resource utilization can be studied for design optimization. More details are provided in Section IV.

The configurable parameters comprise of the following:

1) Fixed Point Data Resolution: We maintain the total resolution length and the fraction length as parameters throughout the system.

2) Number of Input Features

3) Number of Virtual and Physical Hidden NPCs

4) Number of Virtual or Physical Hidden Layers

5) Frequency of the clock

6) Accuracy of the sigmoid unit

The accuracy of sigmoid unit can be made configurable by parameterizing the number of lookups for the piece-wise linear approximation.

Run Time Configurability

Run Time Configurability is of considerable significance. This allows the same hardware to be used for different applications or for the same application but with higher recognition accuracy, effectively allowing cascading of neural networks. Cascading of networks as discussed earlier may be essential for certain types of application using low power sensor nodes. Consider the example of an automatic camera device for recording of wildlife activity in areas where continuous access is difficult [17]. The idea may be to filter the relevant images based on detection of motion. Further we may want to know if the detected animal belongs to an endangered species like a tiger and to immediately send an alert signal with the location coordinates. Such a camera needs to continuously sense

the surroundings for motion and it would be highly inefficient to attempt to classify the animal for all the frames. Also, it is desirable to have the engine responsible for detection of motion also performs the task of classification of the animal. The design provides compile time configurability by using parameterization. For run-time configurability, virtualization proves useful since by simply controlling the number of phases it allows us to implement a bigger network with higher representational power by increasing the number of hidden neurons or the number of hidden layers. The downside being that the storage of such a system needs to equal that of the network with the highest storage requirements. For cascaded networks we need to provide equal to the total of all the cascaded networks.

CHAPTER 4

CASE STUDY: BODY WORN CAMERAS

Our main goal is to study the tradeoffs of power, timing and resource utilization with different network configurations and also the resolution of the data within the network for a specification application case study. We select Body Worn Cameras as recently there has been increased interest for the use of Body Worn Cameras for law enforcement. Initially we also performed experimentation based on the MNIST digit database and obtained accuracy results which matched that of the other research papers. We proceeded to BWCs and posture recognition because of recent surge in interest for surveillance needs. Automatic recognition of human actions and postures is a key enabler for both video surveillance and Body-Worn Cameras.

Adoption of Body Worn Cameras

Body Worn Cameras (BWCs) are gaining traction both commercially and from the law enforcements' point of view. Multiple pilot programs are being conducted for BWCs manuscript including those in Mesa, Arizona, in the United States [18], Plymouth, United Kingdom [19]. These studies have highlighted the potential of such video cameras to capture much more compelling evidence and also act as a deterrent to crime. These also highlight benefits such as increase in accountability and transparency.

Motivation of Selecting BWCs for Case Study

Short battery life, limited storage capacity [20] as well as the need for a human operator to analyze the data, limit the wide-spread adoption of the BWCs. Since data is analyzed off-line, it cannot be used for triggering affirmative action such as alerting law enforcement. To enhance battery life, the current cameras are manually turned on and off, which defeats the purpose of 'always-on' sensing. Fig. 1. (b) and (c) show the power consumption of different sub-components of the camera and provide a comparison between FPGA and ASICs for a continuously wireless transmitting module. It can be noted that of the total power consumption approximately only 25% of the total power is spent in the actual camera and sensing, with the rest being for codecs, storage and transmission. It is thus desirable to enable 'smartness' such that the camera would be able to make intelligent and judicious decisions on when to start storing a video stream while at the same time providing a metric of human aggressiveness in the field of view which corresponds directly to the basis of this thesis. To enable ultra-low power operation with continuous sensing we envision a cascade of classifiers. The front end classifier needs to be able to perform an initial filtering of images based on relevancy. This initial decision regarding relevancy is specific to the application under consideration. For example detection of motion, detection of a human or a human face, etc. This would be then followed by a deeper, power consuming classifier targeting high accuracy. For our chosen case study of Body Worn Cameras we target the detection of human posture. Aggressiveness may be associated with human posture and hence, we propose a hardware assisted camera front-end capable of detecting human posture and identifying relevant

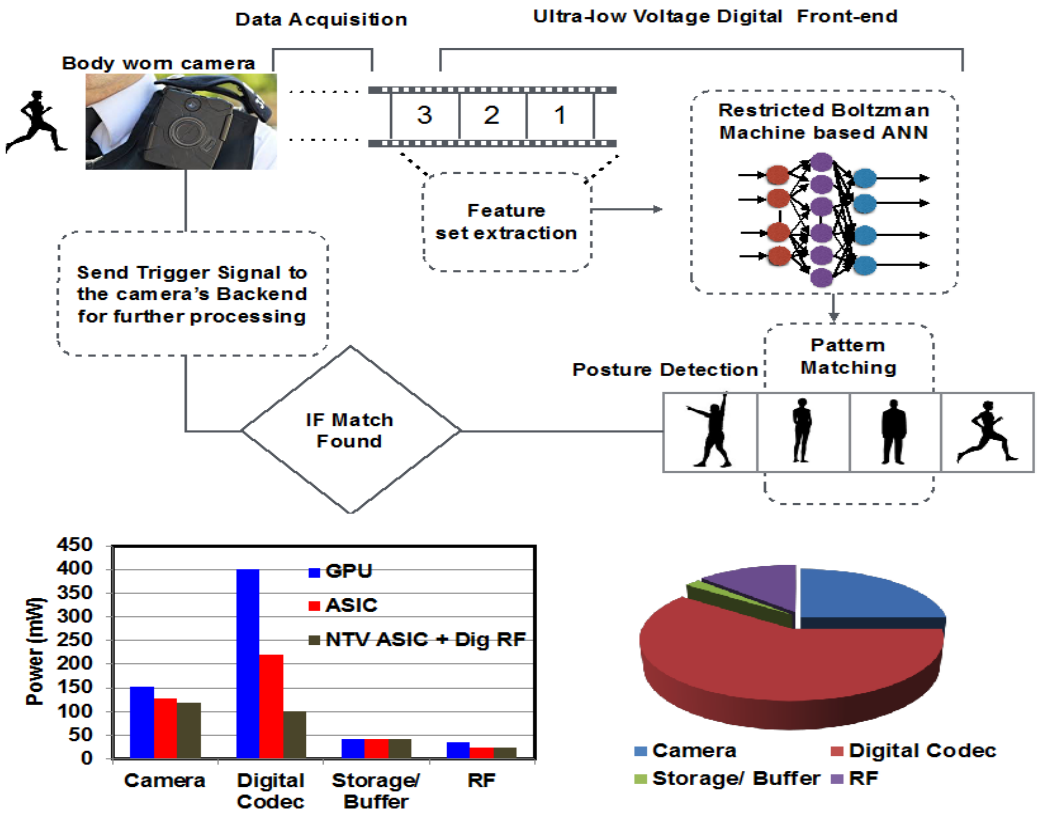


Fig. 7. (a) Usage model for a typical 'always-on' body worn camera (b) The different components of power dissipation in a state-of-the-art camera based sensor node with continuous wireless transmission (c) Breakdown of power illustrating a large section of the total dissipation in the digital codec (H.264).

'information' in incoming video stream. This hardware architecture needs to be co-optimized with the algorithm as well as the frame-rate, data resolution and accuracy targets. When cascaded to the data acquisition (pixel array and analog-to-digital converters) unit, this system can allow ultra-low power video capture as well as intelligent data assimilation. Fig. 7 (a) illustrates the envisioned system.

Image Database for Posture Detection

The experiments are carried out on the Weizmann human silhouette based action database [21] (Fig. 8). The database consists of video sequences (180 x 144, de-interlaced 50 fps) of nine different actors, each performing ten different actions such as “bending”, “jumping-jack”, “jumping forward-on-two-legs”, “jumping-in-place-on-two-legs”, “running”, “galloping-sideways’’, “waving-with-one-hand”, “waving with-two-hands”. To obtain the silhouettes, we perform background subtraction. The algorithm and hardware implementation of such a unit has been described in [22] [23] [24]. These silhouettes are aligned and the training of the neural network is performed using these aligned silhouettes. It is interesting to note that with the popularity and deployment of BWCs, this data base is evolving and better training sets are expected in the recent future. Different postures from the Weizmann database correspond to basic human postures and are applicable to BWCs. For example, ‘putting both hands up’ is treated as a defensive posture while ‘running’ is treated as aggressive behavior. Once proper posture identification is enabled, the output can be used for further action as the situation and usage demands. However, posture identification is a key primitive that can enable ‘always-on’ BWCs for law enforcement. Other primitives such as sound may also be combined with posture identification to provide indications with better confidence.

Fig. 8. (a) Actions in Weizmann Human Actions Silhouette Database (b) 'Both Arms Raised' - Action Silhouette

Experimentation Setup and Methodology

We perform off-line training on sample dataset using MATLAB and then transfer the weights to the Xilinx compiler using "Memory Initialization Files". The training set is divided into mini-batches. The RBM is trained in an unsupervised manner using Contrastive Divergence. The features generated by the RBM are used to train the classifier. Since, our input (pixel data) is real valued and not binary, we scale them to [0, 1] and treat them as probabilities [8]. As per [8] the learning process remains the same. We do not perform back-propagation to further tune the parameters, because it may cause over fitting since the labeled data of the Weizmann database is limited. The trained weights are then extracted and fed into the FPGA platform at compile time as memory initialization files. Classification in contrast to training just involves a forward pass using the hardware system described above after the initial pre-processing step which involves background subtraction as shown in Fig. 9. The baseline Neural Network configuration selected for experimentation is a shallow network comprising of 256 feature inputs, 300

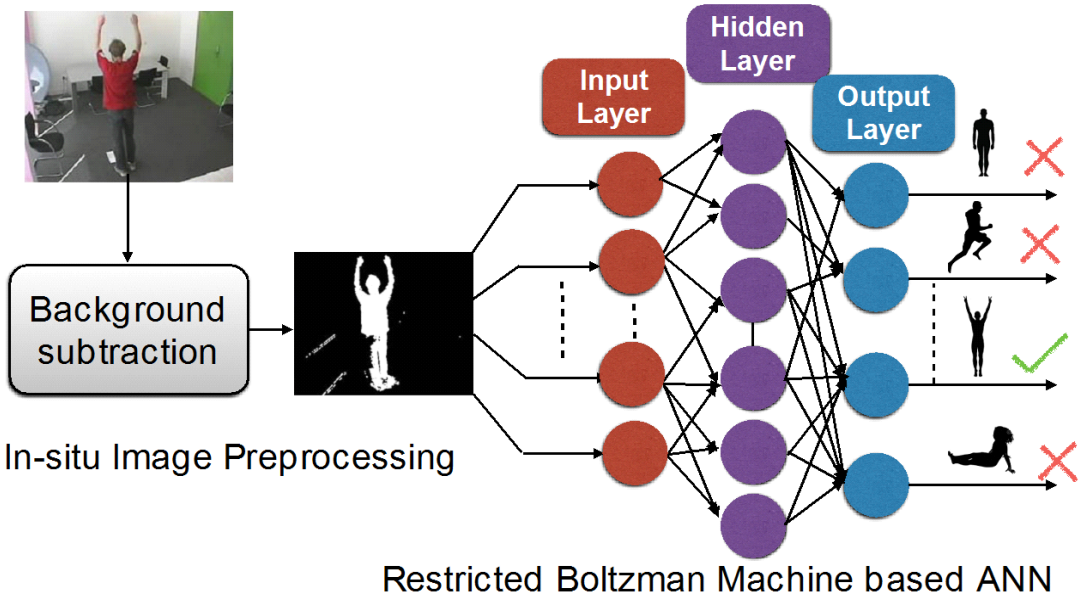


Fig. 9: The recognizer flow highlights the layers of the Restricted Boltzmann Machine and the pre-processing unit comprising of background subtraction. The output layer is designed as winner-take-all and the posture with highest probability is chosen.

virtual NPCs, and 30 physical NPCs for hidden layer 1 and a final layer comprising of 10 NPCs corresponding to 10 silhouette actions.

Algorithmic Accuracy

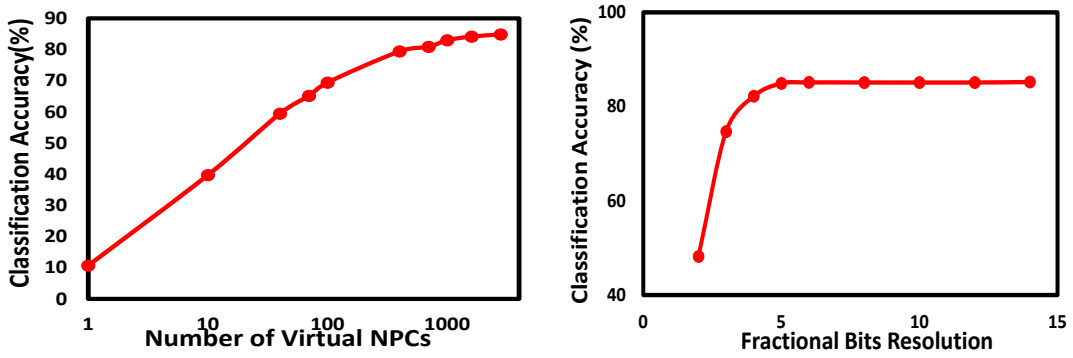
Fig. 10. (a) Illustrates the trade-off between accuracy and the number of virtual NPCs. We observe an increase in accuracy of the network as the number of Virtual NPCs are increased. We note the saturating nature of the curve and for a target accuracy rate of 80% we choose a baseline design with 300 virtual NPCs. Fig. 10. (b) Illustrates the dependence of recognition accuracy on the bit width of the data representation. With a fractional bit width of 6 (Q2.6 format) and avoiding overflow while accumulating, the accuracy tends to that of a floating point representation and has been chosen for our design. This results in lower design complexity and power without compromising the

accuracy of recognition. We also test our network on other standard databases such as MNIST and observe results of more than 94% accuracy which matches the results obtained by other authors who use RBMs for classification.

Fig. 11. Shows the simulation results for running the network on Xilinx's simulator with a phase count of 30.

Hardware Measurement Results

The most important design criteria is the choice of the number of physical NPCs. As seen in Fig. 12 (a) the resource utilization of the network can be improved by reducing the number of physical NPCs. This however results in an increase the in the processing time as shown in Fig. 12 (b). It should, however be noted, that at 30 fps the amount of time available for processing the data is sufficient with a small number of physical NPCs. The choice of the data bit width also has significant impact on the resource utilization of the network and has been shown in Fig. 12 (c), which further justifies the notion of using a low bit width (8 bits here) for data representation.



(a) (b)

Fig. 10. (a) Showcases the increasing in Recognition Accuracy with Increase in Number of Virtual NPCs in the hidden layer of a Network as the network gains more representation power.
(b) Classification Accuracy with varying fixed point representation resolutions. The integer bits kept constant at 1. We make sure there is low probability of overflow by having higher bit width for the accumulator

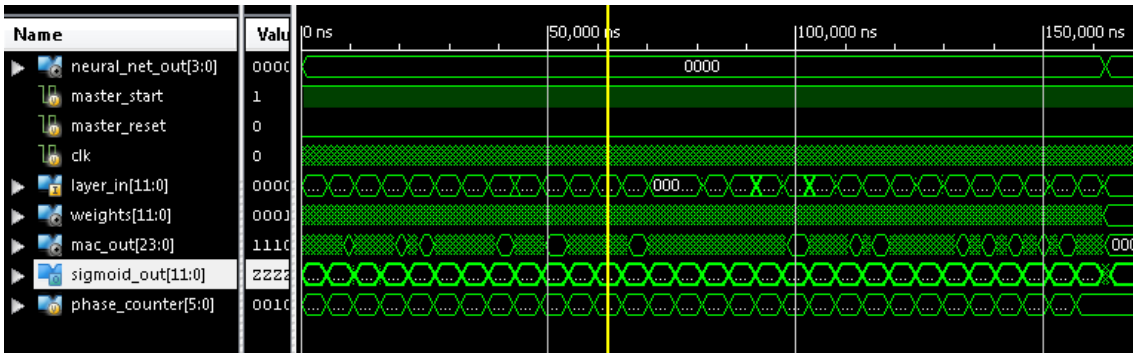


Fig. 11. Shows the Simulator output for a network configuration of 300 Virtual hidden neurons with 10 physical neurons, requiring a total of 30 phases as seen in the diagram. The data-path used for this simulation is of 12 bits. The input to the layer, layer_in, is used for sending the image pixels serially and the same is replayed on every new phase.

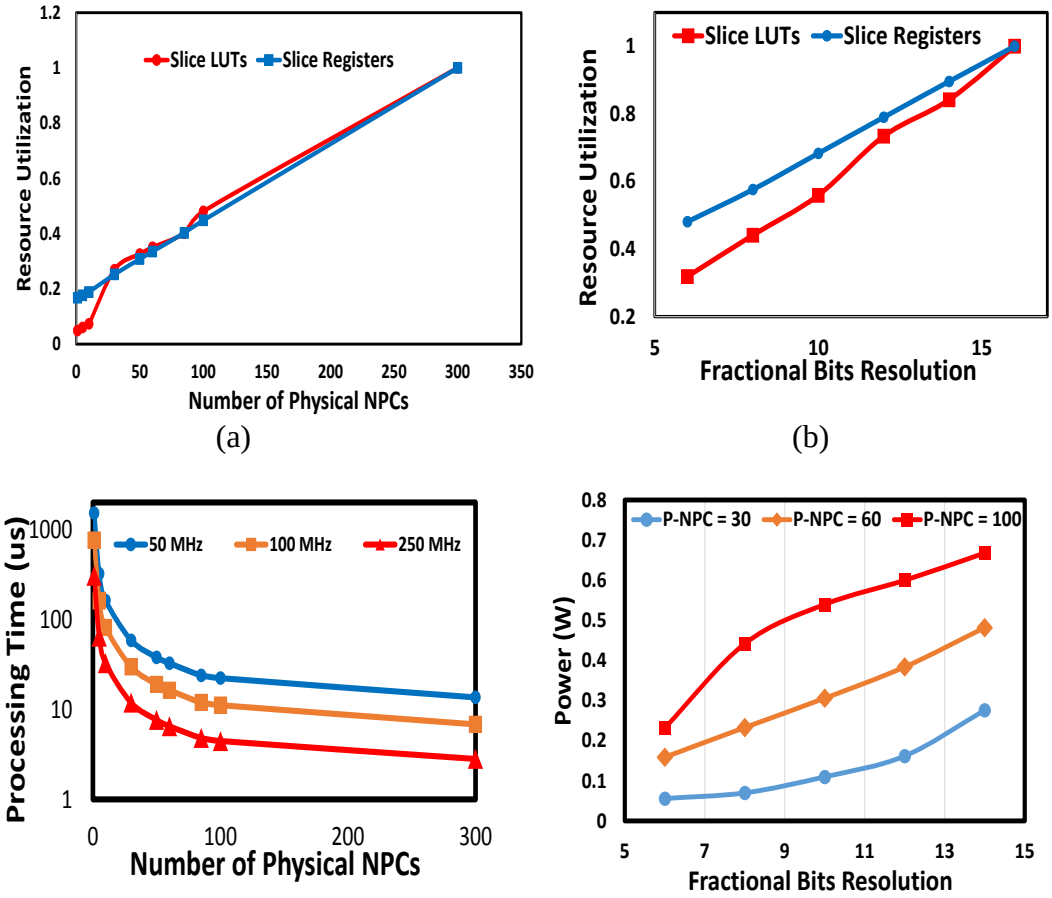


Fig. 12. (a) Describes the Normalized Resource Utilization for increase in parallelization in the network layer. The normalization is with respect to the resource utilization of NPC = 300 (Slice LUT = 97572, Slice Registers = 29582)

(b) Describes the Increase in Processing Time as the parallelization is reduced by reusing the Physical NPCs for computation so as to save resources and power

(c) Normalized Resource Utilization vs. fractional bit resolution. Integer bits kept constant. Normalization is carried with respect resource utilization of 16 bits resolution (Slice LUTs = 35787, Slice Registers = 9437)

(d) Power vs. Fractional bit resolution for different network Physical (P) NPCs. The integer bits are kept constant.

Design Space Exploration

To minimize the overall network power and utilization at acceptable performance, we jointly optimize algorithms and hardware. We use Matlab for our training the neural network and generating the weights. The output results obtained using Matlab also serve as our reference results for ensuring correctness of our FPGA results. It has already been shown that for acceptable performance, we choose 8 bits for data representation. The number of virtual NPCs is chosen as 300. We explore the entire design space of power and the bit width of data representation as a function of the total number of physical NPCs (Fig. 12 (d)). We observe that increasing the number of NPCs increase the total power dissipation but results in faster compute (Fig. 12 (c)). Further, the power increases rapidly with the data width. Finally it is important to understand the impact of serialization to the total energy cost of the design (i.e., the energy required to compute per frame). Fig. 13 illustrates the total energy cost of the design as a function of the number of physical NPCs when operated at 50 MHz For a large number of NPCs, the total (leakage and dynamic) power increases whereas for a small number of NPCs, the total data movement and time to process increases rapidly (Fig. 12 (c)). The point of minimum energy is measured for 30 physical NPCs (with 300 virtual NPCs) as seen in Fig. 13. This illustrates the need for hardware-software co-design & by joint optimization of the accuracy-energy-resource utilization space, an optimum design point is attained. At this design point, we note less than 5nJ of energy/frame for processing. This illustrates three orders of magnitude improvement in total power (<20mW) compared to a camera based wireless sensor node.

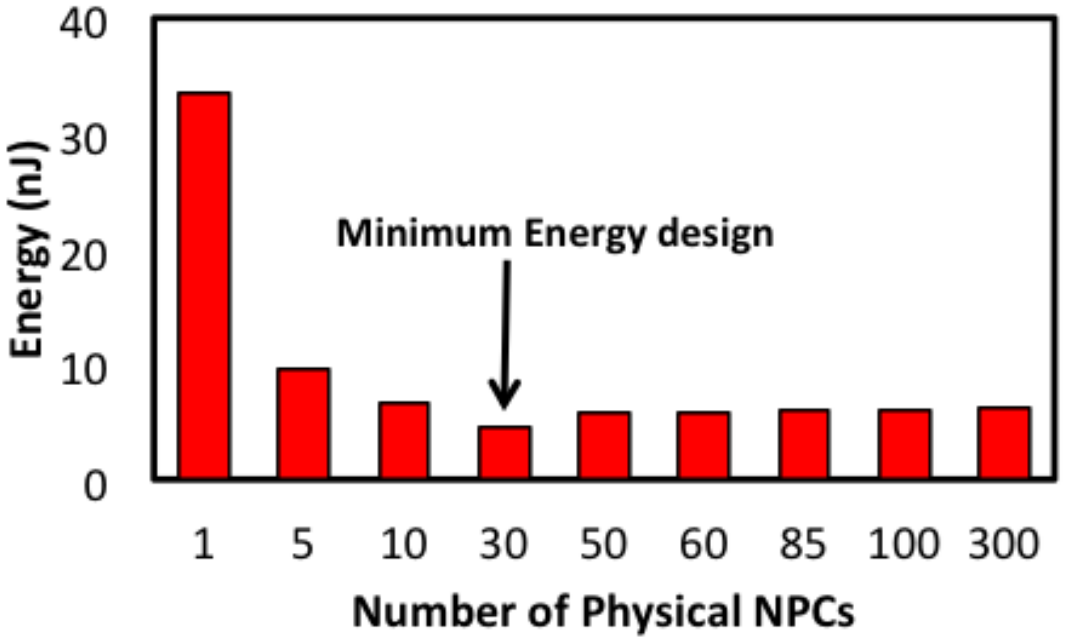


Fig. 13. Total energy/frame as a function of the number of physical NPCs. The 'Minimum Energy' design is obtained for 30 physical NPCs running at a clock frequency of 50 MHz

CHAPTER 6

CONCLUSION AND FUTURE WORK

This thesis describes the design of an RBM based reconfigurable low power hardware engine and presents a case study on ‘human posture’ identification in ‘always-on’ Body-Worn-Cameras. Design space exploration reveals the need for algorithm-hardware co-optimization and illustrates a minimum energy design point for thirty physical NPCs. At the minimum energy point, we spend less than 5nJ per frame and achieve greater than 80% accuracy in posture detection. Below we discuss the future scope of research based on this thesis.

Based on the results obtained for different configurations and the presence of minimal energy design point, a software neural compiler can be built which analyzes the constraints of the input application such as power, processing time, input feature size and outputting the network configuration which is closest to the expectations. An extensive experimentation is required so as to capture the tradeoffs observed allowing the compiler to optimize and generate the right configuration

There are several ways in which power can be further optimized, such as opting for multiple clock domains. Since, we observe a producer-consumer dependency across the computational layers, we can reduce the frequency of clock at the layer with lesser number of computations. The effectiveness of Clock gating, Power gating, Dynamic Voltage Frequency Scaling etc. can also be evaluated for this design.

REFERENCES

[1] Rojas, Raúl. *Neural networks: a systematic introduction*. Springer Science & Business Media, 1996.

[2] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." *Cognitive modeling* 5 (1988).

[3] Yoshua Bengio (2009), "Learning Deep Architectures for AI", Foundations and Trends® in Machine Learning: Vol. 2: No. 1, pp 1-127. http://dx.doi.org/10.1561/2200000006

[4] Hinton, Geoffrey, Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets." *Neural computation* 18.7 (2006): 1527-1554.

[5] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) **ImageNet Large Scale Visual Recognition Challenge**. *arXiv:1409.0575,* 2014

[6] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks"

[7] Bengio, Yoshua, and Yann LeCun. "Scaling learning algorithms towards AI."*Large-scale kernel machines* 34.5 (2007).

[8] Asja Fischer, Christian Igel, "Training Restricted Boltzmann Machines: An Introduction"

[9] G. E. Hinton, "Training products of experts by minimizing contrastive divergence", *Neuron Computation,* 14: 1771-1800, 2002.

[10] Zhu, Jihan, and Peter Sutton. "FPGA implementations of neural networks–a survey of a decade of progress." *Field Programmable Logic and Application*. Springer Berlin Heidelberg, 2003. 1062-1066.

[11] Kim, Sang Kyun, et al. "A highly scalable restricted boltzmann machine FPGA implementation." *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*. IEEE, 2009.

[12] Le Ly, Daniel, and Paul Chow. "High-performance reconfigurable hardware architecture for restricted Boltzmann machines." *Neural Networks, IEEE Transactions on* 21.11 (2010): 1780-1792.

[13] Ly, Daniel L., Volodymyr Paprotski, and Danny Yen. "Neural networks on gpus: Restricted boltzmann machines." *University of Toronto* (2008).

[14] Jiang, Jingfei, et al. "Accuracy evaluation of deep belief networks with fixed-point arithmetic." *Computer Modelling & New Technologies* 6 (2014).

[15] Deka, Biplab. "Towards a Low Power Hardware Accelerator for Deep Neural Networks.".

[16] Gomperts, Alexander, Abhisek Ukil, and Franz Zurfluh. "Development and implementation of parameterized FPGA-based general purpose neural networks for online applications." *Industrial Informatics, IEEE Transactions on* 7.1 (2011): 78-89.

[17] Dodge, Wendell E., and Dana P. Snyder. "An automatic camera device for recording wildlife activity." *The Journal of Wildlife Management* (1960): 340-342.

[18] Police Executive Research Forum (PERF), “Implementing a Body-Worn Camera”

[19] Martin Goodall, Home Office, “Guidance for the Police Use of Body-Worn Video Devices”

[20] Jonathan Hayes, Dr. Lars Ericson, ManTech and NLECTC, A primer on Body-worn cameras for law enforcement.

[21] L. Gorelick, M. Blank, E. Shectman, M. Irani and R. Basri, “Actions as Space-Time Shapes”, IEEE PAMI, vol. 29, no. 12, pp. 2247-2253, 2007.

[22] James W. Davis, Aaron F. Bohick, “A Robust Human-Silhouette Extraction Technique for Interactive Virtual Environments”

[23] Kyungnam Kim, Thanarat H. Chalidabhonse, David Harwood, Larry Davis, “Real-time foreground-background segmentation using codebook model”

[24] Iffat Zafar, Usman Zakir, Ilya Romanenko, Richard M. Jiang, Eran Edirisinghe “Human Silhouette Extraction on FPGAs for Infrared Night Vision Military Surveillance”